Eight HP-12C Games

Tony Hutchins, #1049

The first 7 require a seed in [i]. Re-play does not require a re-seed. None require registers cleared beforehand. The classic seed is .5284163. 7 digits are enough.

1. CRAPS. Set [i] **&** [FV]**. Repeat: bet** [R/S] and see roll and updated balance**. Restart:** [f][PRGM][R/S]. **FV**=0=starting balance for a new shooter. On the **first** throw: If total= 7 or 11('natural') then win else if total= 2 or 3 or 12 ('crap') then lose else total='point' and throw again. On subsequent throws: If total=7 then lose else if total='point' then win else throw again.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[g][LSTx]	28-43 36	4	57- 4
[f] CLEAR [PRGM]	00-	1	29- 1	[g][x≶y]	58-43 34
[PMT]	01- 14	0	30- 0	[g][GTO]61	59-43,33 61
0	02- 0	[÷]	31- 10	[g][GTO]77	60-43,33 77
[PV]	03- 13	[RCL]1	32-45 1	3	61- 3
0	04- 0	[+]	33- 40	[×]	62- 20
[n]	05- 11	1	34- 1	[g][x≶y]	63-43 34
6	06- 6	0	35- 0	[g][GTO]77	64-43,33 77
[RCL][i]	07-45 12	[÷]	36- 10	1	65- 1
9	08- 9	[+]	37- 40	[−]	66- 30
9	09- 9	[g][PSE]	38-43 31	[g][x≶y]	67-43 34
7	10- 7	[RCL][PV]	39-45 13	[g][GTO]75	68-43,33 75
[×]	11- 20	[g][x=0]	40-43 35	4	69- 4
[g][FRAC]	12-43 24	[g][GTO]53	41-43,33 53	[−]	70- 30
[i]	13- 12	[RCL]3	42-45 3	[−]	71- 30
[×]	14- 20	7	43- 7	[g][x=0]	72-43 35
1	15- 1	[−]	44- 30	[g][GTO]75	73-43,33 75
[+]	16- 40	[g][x=0]	45-43 35	[g][GTO]04	74-43,33 04
[g][INTG]	17-43 25	[g][GTO]77	46-43,33 77	[RCL][PMT]	75-45 14
[g][CFj]	18-43 14	[RCL]3	47-45 3	[g][GTO]79	76-43,33 79
[RCL][n]	19-45 11	[RCL]0	48-45 0	[RCL][PMT]	77-45 14
2	20- 2	[−]	49- 30	[CHS]	78- 16
[g][x≶y]	21-43 34	[g][x=0]	50-43 35	[RCL][FV]	79-45 15
[g][GTO]24	22-43,33	[g][GTO]75	51-43,33 75	[+]	80- 40
[g][GTO]06	23-43,33	[g][GTO]04	52-43,33 04	[FV]	81- 15
[RCL]1	24-45 1	1	53- 1	[g][GTO]00	82-43,33 00
[RCL]2	25-45 2	[PV]	54- 13	[f][P/R]	
[+]	26- 40	[RCL]3	55-45 3		
[STO]3	27-44 3	[STO]0	56-44 0		

2. DEALER/LOTTO. Set i, PV & FV**.** R/S**. Wait.** RCL 1, RCL 2, ... or optionally store modulus in PMT and RCL 1 R/S, RCL 2 R/S ... **Restart:** f PRGM R/S. **FV** = sample size (max=12), **PV** = Population size. PV>=FV

Selects items ***without replacement*** from a numerical deck or bin, numbered 1,2,3... up to PV('population value'). No concept of shuffle is needed. Not really a game but you can deal poker hands (PV=52 and modulus=13 or PV=53 with joker considered as the first card of the fifth suit=card 5.1), play bingo or even choose tarot cards. Or, best of all you can choose **Lotto** numbers - here we use PV=40 and FV=6 for that, and just RCL 1, RCL 2 ... RCL 6 to see the numbers. For lotto line 39 can just be GTO 00 and we don't need lines 41-57. So, in only 40 lines we have a 12C Lotto generator.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
f P/R		g GTO 32	19-43,33 32	g GTO 41	39-43,33 41
f CLEAR PRGM	00-	RCL g CFj	20-45,43 14	g GTO 03	40-43,33 03
0	01- 0	RCL PMT	21-45 14	R/S	41- 31
n	02- 11	−	22- 30	1	42- 1
RCL PV	03-45 13	g x=0	23-43 35	−	43- 30
RCL i	04-45 12	g GTO 29	24-43,33 29	RCL PMT	44-45 14
9	05- 9	RCL n	25-45 11	÷	45- 10
9	06- 9	g x=0	26-43 35	g INTG	46-43 25
7	07- 7	g GTO 32	27-43,33 32	1	47- 1
×	08- 20	g GTO 20	28-43,33 20	g LSTx	48-43 36
g FRAC	09-43 24	RCL 0	29-45 0	g FRAC	49-43 24
i	10- 12	n	30- 11	RCL PMT	50-45 14
×	11- 20	g GTO 03	31-43,33 03	×	51- 20
g INTG	12-43 25	RCL 0	32-45 0	1	52- 1
1	13- 1	n	33- 11	+	53- 40
+	14- 40	RCL PMT	34-45 14	%	54- 25
PMT	15- 14	g CFj	35-43 14	+	55- 40
RCL n	16-45 11	RCL n	36-45 11	+	56- 40
STO 0	17-44 0	RCL FV	37-45 15	g GTO 41	57-43,33 41
g x=0	18-43 35	g x≤y	38-43 34	f P/R	

3. JIVE TURKEY. Set i & FV**.** R/S**. Repeat: Guess**(0-99)**,** R/S, seeing plus 1 if you are high and -1 if you are low, and your number of guesses if you correctly guess the secret number**. Restart:** f PRGM R/S. **FV**= %Truth (0-100).

Originally by Maurice E.T.Swinnen founder and Editor of TI PPC Notes (1980-1982). Transcribed for HP-67 in PPC Journal Jan 1978 by Terry Mickelson(PPC #1926). The HP-67 has flags and subroutines. Here we use FV as a flag and replicate the 63 line HP-67 version in 47 lines.

You won't know what a +1 means after your first guess. Just re-run with the same guess and if you were right you'll see a 2 :-)

i or FV can be changed at any time. Lower FV if you are getting jived too much. FV=0 is nice and simple - then your false values will indeed be zero. :-) Note the %T at line 21 is the %Truth calculation. This is misleading :-).

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
f P/R		7	16- 7	PV	33- 13
f CLEAR PRGM	00-	×	17- 20	RCL 1	34-45 1
0	01- 0	g FRAC	18-43 24	RCL 2	35-45 2
STO 0	02-44 0	i	19- 12	−	36- 30
PV	03- 13	ENTER	20- 36	g x=0	37-43 35
g GTO 13	04-43,33 13	%T	21- 23	g GTO 07	38-43,33 07
R↓	05- 33	×	22- 20	g LSTx	39-43 36
STO 1	06-44 1	g INTG	23-43 25	RCL 1	40-45 1
RCL 0	07-45 0	RCL PV	24-45 13	g x≤y	41-43 34
R/S	08- 31	g x=0	25-43 35	g GTO 46	42-43,33 46
STO 2	09-44 2	g GTO 05	26-43,33 05	RCL PV	43-45 13
1	10- 1	R↓	27- 33	CHS	44- 16
STO + 0	11-44 40 0	RCL FV	28-45 15	g GTO 08	45-43,33 08
PV	12- 13	g x≤y	29-43 34	RCL PV	46-45 13
RCL i	13-45 12	g GTO 34	30-43,33 34	g GTO 08	47-43,33 08
9	14- 9	1	31- 1	f P/R	
9	15- 9	CHS	32- 16		

4. SUPER BAGELS. Set i **&** FV**, base** R/S→FV.0 (your target)**. Repeat: Guess** (an integer of FV digits)**,** R/S and see FERMI.PICO, until you win (FERMI=FV, PICO=0). **Restart:** f PRGM **base** R/S. BTW the HP-67 Games Pac version of this is 223 lines! **FV**= The number of digits in the secret integer (1-6 here, 1-8 on the 67 version). Your mission is of course to guess the secret integer. Before pressing R/S to start a game you *must* specify the secret number **base**. 10 R/S uses decimal - i.e. all digits from 0-9. 8 R/S uses octal, 2 R/S uses binary, etc. FERMI is the number of direct hits, and PICO the wayward hits. The direct/wayward hits are guess digits in the correct/wrong place in the secret number. FERMI.PICO=0.0 means we have 4 BAGLES! :-).

Example 1: Let's run with .5284163 in i and the usual 4 decimal digits: 4 FV f PRGM 10 R/S→4.0 (your target), and make 5284 our first candidate.

5284 R/S →0.3	2580 R/S →2.2	8520 R/S →4.0	Done!

Example 2: f PRGM 10 R/S→4.0

1234 R/S→1.0	5678 R/S→0.2	9178 R/S→0.4	1978 R/S→1.3
1798 R/S→2.2	1789 R/S→4.0	Ahha! The French Revolution :-)	

Example 3: [f][PRGM] 10 [R/S]→4.0

5667 [R/S]→2.1	6678 [R/S]→2.1	6617 [R/S]→3.0	6627 [R/S]→4.0

This code uses *both* CFj & Nj as variable length (j=1 to 6) vectors, which makes it

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[X]	33- 20	[RCL]0	67-45 0
[f]CLEAR[PRGM]	00-	[PMT]	34- 14	[n]	68- 11
[g][CFo]	01-43 13	[RCL]0	35-45 0	1	69- 1
[RCL]0	02-45 0	[n]	36- 11	[g][Nj]	70-43 15
[RCL][i]	03-45 12	[RCL][g][Nj]	37-45,43 15	[R↓]	71- 33
9	04- 9	[RCL][g][CFj]	38-45,43 14	[g][x=0]	72-43 35
9	05- 9	[RCL][PMT]	39-45 14	[g][GTO]75	73-43,33 75
7	06- 7	[−]	40- 30	[g][GTO]60	74-43,33 60
[X]	07- 20	[g][x=0]	41-43 35	1	75- 1
[g][FRAC]	08-43 24	[g][GTO]66	42-43,33 66	[STO][−]0	76-44 30 0
[i]	09- 12	[RCL][FV]	43-45 15	[RCL]0	77-45 0
[X]	10- 20	[n]	44- 11	[g][x=0]	78-43 35
[g][INTG]	11-43 25	[RCL][n]	45-45 11	[g][GTO]81	79-43,33 81
[g][CFj]	12-43 14	[RCL][g][CFj]	46-45,43 14	[g][GTO]25	80-43,33 25
0	13- 0	[RCL][PMT]	47-45 14	[RCL][FV]	81-45 15
[g][Nj]	14-43 15	[−]	48- 30	[n]	82- 11
[RCL][n]	15-45 11	[g][x=0]	49-43 35	0	83- 0
[RCL][FV]	16-45 15	[g][GTO]55	50-43,33 55	[RCL][g][Nj]	84-45,43 15
[g][x≤y]	17-43 34	[RCL][n]	51-45 11	[g][x=0]	85-43 35
[g][GTO]21	18-43,33 21	[g][x=0]	52-43 35	[g][GTO]88	86-43,33 88
[g][GTO]02	19-43,33 02	[g][GTO]75	53-43,33 75	[1/x]	87- 22
[R↓]	20- 33	[g][GTO]45	54-43,33 45	[+]	88- 40
[R/S]	21- 31	[R↓]	55- 33	0	89- 0
[PV]	22- 13	[n]	56- 11	[g][Nj]	90-43 15
[RCL][FV]	23-45 15	[RCL][g][Nj]	57-45,43 15	[R↓]	91- 33
[STO]0	24-44 0	[g][x=0]	58-43 35	[RCL][g][CFj]	92-45,43 14
1	25- 1	[g][GTO]62	59-43,33 62	[R↓]	93- 33
0	26- 0	[RCL][g][CFj]	60-45,43 14	[RCL][n]	94-45 11
[RCL][PV]	27-45 13	[g][GTO]51	61-43,33 51	[g][x=0]	95-43 35
[%]	28- 25	1	62- 1	[g][GTO]20	96-43,33 20
[g][INTG]	29-43 25	0	63- 0	[R↓]	97- 33
[PV]	30- 13	[g][Nj]	64-43 15	[g][GTO]84	98-43,33 84
[g][LSTx]	31-43 36	[g][GTO]75	65-43,33 75	[f][P/R]	
[g][FRAC]	32-43 24	[R↓]	66- 33	**FERMI + PICO ≤ FV**	

easily the most complex game here. Each guess digit is processed just the once, but the backward jump at line 74 can initiate some implicit secondary processing. The 5667 [R/S]→2.1 above illustrates this, where the 2 sixes contribute "1.1" :-)

5. SUM OF DIGITS (SOD). Set [i]. [R/S] see initial SOD(0-18) of the secret number (0-99)**. Repeat: Guess** to be added to secret number**,** [R/S], seeing 0.000000000 if you are over 99, in which case guess is not added, 99.00000000 if you win, and the SOD of the new secret number otherwise. **Restart:** [f][PRGM][R/S]

Thanks to Gene Wright who pointed this game out to me - it is from page 27 of HP Digest number 5, 1979. There a 41 line program was used, for the HP33-E, requiring 4 registers. Here we use only [i] and R_0, and 36 lines :-) This, our third "secret number" game, is the shortest of all the games here, and quite delightful to play! The random number generator using the 997 and the .5284163 seed is taken from the "HP-12C Solutions Handbook". See the Fruit Machine game for the simplest seed :-)

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[STO]0	12-44 0	[g][x≤y]	25-43 34
[f]CLEAR[PRGM]	00-	1	13- 1	[g][GTO]29	26-43,33 29
[EEX]	01- 26	0	14- 0	[R↓]	27- 33
[RCL][i]	02-45 12	[%]	15- 25	[g][GTO]11	28-43,33 11
9	03- 9	[g][INTG]	16-43 25	[f]9	29-42 9
9	04- 9	9	17- 9	[−]	30- 30
7	05- 7	[X]	18- 20	[g][x=0]	31-43 35
[X]	06- 20	[−]	19- 30	[g][GTO]35	32-43,33 35
[g][FRAC]	07-43 24	[R/S]	20- 31	[CLx]	33- 35
[i]	08- 12	[RCL]0	21-45 0	[g][GTO]20	34-43,33 20
[%T]	09- 23	[+]	22- 40	[g][LSTx]	35-43 36
[g][INTG]	10-43 25	9	23- 9	[g][GTO]00	36-43,33 00
[f]0	11-42 0	9	24- 9	[f][P/R]	

Example 1: .5284163[i][R/S]→11. The secret number is one of eight: 29,38,47,56,65,74,83 or 92. We don't know which. The "safest" addition is 7. Note all possible answers have SOD=7 (7,16,25 etc.). 7 [R/S]→9. There are 10 possible numbers with SOD=9: 9,18,27,36,45,54,63,72,81 and 90. Taking 7 off these we get 2,11,20,**29**,**38**,**47**,**56**,**65**,**74** and **83**. We still have a choice of 7! 9[R/S]→99.0000000 What a fluke!!!! So the original was 83.

Example 2: [R/S]→10. Again a choice of 8 (19,28,...,91). Lets be bold and try not 8 but something random, say 50[R/S]→0.000000000. So, the number is over 49. It must be one of 55,64,73,82 or 91. 26[R/S]→9. We have now boosted our number to 81 or 90. 18[R/S]→99.0000000. Yeah! This is fun, isn't it?

Example 3: [R/S]→9. Now a choice of 9. 50[R/S]→14. 22[R/S]→99.0000000. Another sheer fluke. I'm starting a strategy. 22 has SOD 4, and 4+14=18. The 22 itself was just a guess as the 14 meant the number must be up to over 59 at least. So, possibly 40 to go - and 22 is roughly at the midpoint. Can you tell I've never played this game before?

Check out the code that replaces a 2 digit number with the sum of its digits: 10[%] [g][INTG] 9 [X] [−]. (Lines 13-19). That was the key to shrinking this one :-)

6. PONTOON (Vingt et un or 21 or Blackjack). Set i**.** R/S see 0.00**. Repeat:** R/S, seeing new total score. Negative score=bust. Stop R/S or hitting when desired**.** Simulate any desired banker strategy by restarting**. Restart:** f PRGM R/S

Here we use only i, R_0 and R_1.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
f P/R		g x=0	16-43 35	2	33- 2
f CLEAR PRGM	00-	g GTO 25	17-43,33 25	g x≤y	34-43 34
CLx	01- 35	9	18- 9	g GTO 40	35-43,33 40
STO 0	02-44 0	g x≤y	19-43 34	+	36- 40
STO 1	03-44 1	g GTO 22	20-43,33 22	2	37- 2
R/S	04- 31	R↓	21- 33	−	38- 30
1	05- 1	1	22- 1	g GTO 04	39-43,33 04
3	06- 3	+	23- 40	2	40- 2
RCL i	07-45 12	g GTO 27	24-43,33 27	1	41- 1
9	08- 9	1	25- 1	RCL 0	42-45 0
9	09- 9	STO 1	26-44 1	g x≤y	43-43 34
7	10- 7	STO + 0	27-44 40 0	CHS	44- 16
×	11- 20	RCL 1	28-45 1	CHS	45- 16
g FRAC	12-43 24	g x=0	29-43 35	g GTO 04	46-43,33 04
i	13- 12	g GTO 40	30-43,33 40	f P/R	
×	14- 20	RCL 0	31-45 0		
g INTG	15-43 25	1	32- 1		

Example 1: .5284163 i R/S →0. R/S →10. R/S →18. Close enough to 21 for me. Let's imagine a banker who holds after 16: f PRGM R/S R/S →4. R/S →15 R/S →18. A draw? Let's keep going hoping for something interesting - a few aces at least.

Example 2: f PRGM R/S R/S →10. R/S →20. f PRGM R/S R/S →10. R/S →19. I'd say we win this one. Still no aces though.

Example 3: f PRGM R/S R/S →9. R/S →12. R/S →-22. Bust! So many 10's. All 10's and court cards count as 10 (see lines 18-20 above).

Example 4: f PRGM R/S R/S →7. R/S →17. R/S →18. Ahha at last we got an ace. Too bad it's not much use. One more hit: R/S →-22. Too daring<G>.

Example 5: f PRGM R/S R/S →7. R/S →17. R/S →-27.

Example 6: f PRGM R/S R/S →10. R/S →19. f PRGM R/S R/S →10. R/S →17. Another win. But I don't think I'd do any good in a casino :-) This program is really just a 21 scorer - no betting is incorporated but we get the feel of the game.

Lines 31-39 are dedicated to counting an ace as 11 if doing so prevents bust.

Lines 40-45 just change the sign of the score if it is a bust.

R_1 =1 means at least one ace has been drawn. It is just a flag.

R_0 holds the score, assuming aces count as 1.

7. FRUIT MACHINE. Set i **&**R_1(bank,e.g. \$10)**. Repeat:** R/S →next win. To view bank just RCL **1. Restart:** Just R/S <G>. Here again we use only i, R_0 and R_1. Wins are 0.**000** (\$10 jackpot) and 18 others worth \$1 (0.**111**, 0.**222** etc and 0.**110**, 0.**220** etc). Lines 1-4 deduct \$0.01 per turn, so it's a "penny slot". For \$10=1,000 turns the expected 12C payout is 1·\$10 +18·\$1=\$28, a 180% payout ratio! The waiting time (WT∩***geometric*** with p=.019) between wins averages (1-p)/p=52 spins (1m:45s on 12C), so to avoid ruining the R/S key, the 12C runs till the ***next*** win. Starting with i=.5284163 it takes 3m:25s till the first win, 0.111, after WT=103 "spins". 13.6% ($(1-.019)^{104} \approx e^{-2}$) of WT are ***over*** 104 spins. For the full 500,000 spin cycle, WT ranges from 1 to 568 (i=.9985843 takes 19 minutes!), and we get 19·500=9500 wins (worth \$9,000 or a lot of gum<G>). The gum label above dates from 1906 and is called the "BAR" (like our 0) even on modern slot machines.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
f P/R		f RND	13-42 14	1	27- 1
f CLEAR PRGM	00-	g FRAC	14-43 24	÷	28- 10
•	01- 48	STO 0	15-44 0	g FRAC	29-43 24
0	02- 0	g x=0	16-43 35	×	30- 20
1	03- 1	g GTO 34	17-43,33 34	g x=0	31-43 35
STO − 1	04-44 30 1	•	18- 48	g GTO 36	32-43,33 36
RCL i	05-45 12	1	19- 1	g GTO 01	33-43,33 01
9	06- 9	1	20- 1	9	34- 9
9	07- 9	1	21- 1	STO + 1	35-44 40 1
7	08- 7	÷	22- 10	1	36- 1
×	09- 20	g FRAC	23-43 24	STO + 1	37-44 40 1
g FRAC	10-43 24	RCL 0	24-45 0	RCL 0	38-45 0
i	11- 12	•	25- 48	g GTO 00	39-43,33 00
f 3	12-42 3	1	26- 1	f P/R	

Thanks to Gene Wright who pointed me to the **49 line** HP-25 slot machine program using **8 registers** in the Feb 1976 PPC Journal V3N2P16. The technique in lines 18-31 above is they key to the condensed code here. The FRAC at line 14 is needed as the previous RND may give **1**.000 and that .000 supplies half our jackpots (the other half come from seeds under .005). Storing the constants in registers makes no discernable difference to the speed. To stop at ***every*** spin just ***change line 33*** to g GTO 38. On the ***new*** 12cp this runs 2.3 X faster, but on the ***first*** 12cp it is 3.7 X faster! ***Hint:*** 3E-7 is one of the 500,000 seeds generated when we start with i=.5284163.

$(1-.019)^n$ is an FV with %i=-1.9, PV=-1 and PMT=0. The geometric approaches the continuous ***exponential*** distribution - with "lack of memory", skewness of 2, kurtosis of 6 (hardly bell shaped<G>) and coefficient of variation=1(standard deviation=mean). The call option value for strike price K would be just $P \cdot e^{-K/P}$, if P were the mean price of an "exponential" share - nice and simple!

8. And one of skill: **LUNAR LANDER** [R/S] see VV.0hhh=-50.0500, Fuel=60,"3-2-1-0" **Repeat: Interrupt** at the "0", **Burn**, [R/S], till crash or land. **Restart:** [f][PRGM][R/S] Refer to **Kalevipoeg's** "Moon Landing Simulator" posted on www.hpmuseum.org on 4 Oct 2005. This one calculates the correct crash velocity. You are 500 feet (hhh) descending at 50 ft/sec(VV) with 60 units of fuel to burn. Careful with it! It's not supposed to be a "Lunar Crater Maker" :-) Only 4 registers are used: R_1=h, R_2=V, R_3=fuel and R_4=acceleration.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[RCL]2	25-45 2	[RCL]2	51-45 2
[f]CLEAR[PRGM]	00-	[g][LSTx]	26-43 36	[RCL]4	52-45 4
5	01- 5	[+]	27- 40	[STO][+]2	53-44 40 2
0	02- 0	[f]4	28-42 4	2	54- 2
0	03- 0	[g][PSE]	29-43 31	[÷]	55- 10
[STO]1	04-44 1	[g][PSE]	30-43 31	[+]	56- 40
5	05- 5	[RCL]3	31-45 3	[STO][+]1	57-44 40 1
0	06- 0	[f]0	32-42 0	0	58- 0
[CHS]	07- 16	[g][PSE]	33-43 31	[RCL]1	59-45 1
[STO]2	08-44 2	[g][x=0]	34-43 35	[g][x≤y]	60-43 34
6	09- 6	[g][GTO]63	35-43,33 63	[g][GTO]63	61-43,33 63
0	10- 0	3	36- 3	[g][GTO]12	62-43,33 12
[STO]3	11-44 3	[g][PSE]	37-43 31	[RCL]2	63-45 2
5	12- 5	2	38- 2	[ENTER]	64- 36
[CHS]	13- 16	[g][PSE]	39-43 31	[×]	65- 20
[STO]4	14-44 4	1	40- 1	[RCL]4	66-45 4
1	15- 1	[g][PSE]	41-43 31	[RCL]1	67-45 1
[CHS]	16- 16	0	42- 0	[×]	68- 20
[RCL]1	17-45 1	[g][PSE]	43-43 31	[ENTER]	69- 36
[%]	18- 25	[RCL]3	44-45 3	[+]	70- 40
[%]	19- 25	[g][x≤y]	45-43 34	[−]	71- 30
[RCL]2	20-45 2	[x≷y]	46- 34	[g][√x]	72-43 21
[x≷y]	21- 34	[R↓]	47- 33	[CHS]	73- 16
[−]	22- 30	[STO][−]3	48-44 30 3	[g][PSE]	74-43 31
[g][x≤y]	23-43 34	[STO][+]4	49-44 40 4	[g][GTO]74	75-43,33 74
[g][GTO]28	24-43,33 28	[STO][+]4	50-44 40 4	[f][P/R]	

The longest trip and worst crash is achieved by burning all fuel on the first cycle. We then crash at v=-96ft/sec and the free fall time is 32 seconds ([RCL]2 [−] [RCL]4 [÷]→32). I did manage one perfect landing with "0" flashing and V=h=0.00 :-). It took a 14 burn sequence: 1, 1, 2, 3, 3, 4, 6, 6, 7, 8, 9, 2, 3 and 5. You can force iterative free-fall when all fuel is gone by pressing 1 when you see the fuel status.

The accompanying platg5.pdf contains an annotated platinum listing of games 1-4 and 8: all five fit the new 12 c platinum. Bagels is one line longer. Find out why ;-).